



Programación Python para BigData

Lección 7: Apache Spark con PySpark [1/2]

Spark Streaming y teoría de Grafos

ACTIVIDAD LECCIÓN 7

Objetivos

- | Realizar una pequeña memoria explicando la teoría de Grafos y sus utilidades en Big Data.
- | Realizar un script usando Spark Streaming

Contenido correspondiente a lección 7:

1. Spark y su ecosistema.
2. Spark SQL y DataFrame.
3. Spark RDD

Actividad relacionada con la lección 7:

El alumno deberá entregar tanto un manual (pdf) como enviar un script (.ipynb) como actividad.

Teoría de Grafos**(5,0 ptos)**

El alumno entregará un manual donde explicará en que se basa la teoría de Grafos y su aplicación principalmente en Big data.

Actividad Spark Streaming**(5,0 ptos)**

El alumno realizará una actividad usando Spark Streaming.

Spark Streaming sirve para procesar datos a tiempo real. Para ello seguirá los siguientes pasos:

Será necesario crear los contextos para Spark con Spark streaming usando las siguiente líneas:

```
# Importar las librerías:
from pyspark import SparkConf, SparkContext
from pyspark.streaming import StreamingContext

# crea una configuración spark
conf = SparkConf()
conf.setAppName("StreamApp")

# crea un contexto spark con la configuración anterior
sc = SparkContext(conf=conf)
sc.setLogLevel("WARN")

# crea el Contexto Streaming desde el contexto spark visto arriba con intervalo
de 2 segundos
ssc = StreamingContext(sc, 2)

# establece un punto de control para permitir la recuperación de RDD
ssc.checkpoint("checkpoint_App")
```


Una vez creado el contexto para Streaming crearemos el DStream que gestionaremos como RDD o SQL con ayuda de la guía de Spark:

<https://spark.apache.org/docs/latest/streaming-programming-guide.html>

Existen múltiples ejemplos que podrás emplear como analizar tweets a tiempo real, o un dataset cualquiera, etc., escoge un ejemplo de ellos.

Al final para que se ejecute será necesario comenzar la transmisión de los datos:

```
# comienza la computación de streaming
ssc.start()
# espera que la transmisión termine
ssc.awaitTermination()
# para la transmisión cuando termine
ssc.stop()
```

 **Nota:** En la mayoría de los ejemplos se crea un socket dónde se enviarán los datos a un IP (localhost) y a un puerto (9009), en este caso deberás de crear dos scripts uno donde generarás la recogida de los datos y otro donde te conectarás con spark streaming a ese socket para recoger y procesar los datos. Se usa esta opción enviará los dos archivos .ipynb para poder ser corregida la actividad.

```
# Crear el socket
TCP_IP = "localhost"
TCP_PORT = 9009
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((TCP_IP, TCP_PORT))
s.listen(1)
print("Waiting for TCP connection...")
conn, addr = s.accept()
print("Connected... Starting getting tweets.")

# lee data del puerto 9009
dataStream = ssc.socketTextStream("localhost", 9009)
```

NOTA FINAL: 10 PTOS